Synthetic-to-real domain adaptation for lane detection

Noa Garnett, Roy Uziel, Netalee Efrat, and Dan Levi

General Motors Technical Center Israel - R&D Labs

Abstract. Accurate lane detection, a crucial enabler for autonomous driving, currently relies on obtaining a large and diverse labeled training dataset. In this work, we explore learning from abundant, randomly generated synthetic data, together with unlabeled or partially labeled target domain data, instead. Randomly generated synthetic data has the advantage of controlled variability in the lane geometry and lighting, but it is limited in terms of photo-realism. This poses the challenge of adapting models learned on the unrealistic synthetic domain to real images. To this end we develop a novel autoencoder-based approach that uses synthetic labels unaligned with particular images for adapting to target domain data. In addition, we explore existing domain adaptation approaches, such as image translation and self-supervision, and adjust them to the lane detection task. We test all approaches in the unsupervised domain adaptation setting in which no target domain labels are available and in the semi-supervised setting in which a small portion of the target images are labeled. In extensive experiments using three different datasets, we demonstrate the possibility to save costly target domain labeling efforts. For example, using our proposed autoencoder approach on the llamas and tuSimple lane datasets, we can almost recover the fully supervised accuracy with only 10% of the labeled data. In addition, our autoencoder approach outperforms all other methods in the semi-supervised domain adaptation scenario.

1 Introduction

Accurate lane detection is a critical enabler for autonomous driving, serving as a primary input for the path planning stage. All current state-of-the-art implementations involve training a single-frame CNN based detector [1–3]. The real-world success of the resulting detector relies on the assumption that the training set reliably represents the operational conditions. This poses the challenge of collecting and labeling images that represent all possible driving scenarios: from highway to urban scenes, in all weather and lighting conditions, covering all lane marking types from all the different geographic regions. Labeling such a large and diverse corpus of data is a highly demanding task to say the least. While several highly-diverse on-road datasets were collected and made public, many of them lack lane annotations (e.g. Waymo open dataset [4], nuScenes [5]). Synthetic datasets have also been proposed for training autonomous perception

models [6],[7]. Unfortunately, the scene creation is still a manual labor demanding task, but more importantly, lanes are often not inserted as graphical objects but as road texture, and therefore no lane annotation can be automatically extracted.

Recently, [2] introduced a methodology for randomly and efficiently generating synthetic data with lane annotation. This approach has the advantage of generating scenes with high variability in lane topology and 3D geometry, without requiring any manual labor. The caveat lies in the limited variability in the appearance of the lanes and the surrounding scene, and generally in the photo-realism of the resulting images, as can be seen in Figure 1(a). We propose to leverage this synthetic data generation method, along with unlabeled real-world data, and to perform synthetic-to-real domain adaptation (DA) in order to overcome the appearance gap. This would allow exploiting the variability in lane geometry provided by the synthetic data along with the appearance variability of unlabeled real data.

Although domain adaptation is a well-studied field, it has yet to be applied to lane detection. In this work, we introduce a novel autoencoder-based approach and a new self-supervision objective addressing domain adaptation for the lane detection task. In addition, we explore existing domain adaptation methods by adjusting them to lane detection. We address domain adaptation in two different settings: unsupervised and semi-supervised. In the Unsupervised Domain Adaptation (UDA) setting, training is done with only source domain labeled data and target domain unlabeled data. In the Semi-Supervised Domain Adaptation (SSDA) setting training has additionally access to a small set of labeled examples from the target domain. In both settings, the goal is to learn an accurate model for the target domain, with minimal compromise compared to a fully supervised model.

Our proposed autoencoder-based method, is inspired by a recent study [8], showing that it is feasible to learn human landmark detection from only unlabeled images and unpaired labels. Similarly, we introduce a method that learns to detect lanes using unlabeled images, and a set of ground truth "logical lane images" (Figure 2(b)), which are not paired with input images. This allows to train a lane detector without even having rendered synthetic data, but with only "logical" top view images of valid lane markings. Based on the assumption that there is a strong correlation between the lanes in a scene and the gradients in the image, we train an autoencoder of the image gradients that passes through a "lane image" bottleneck. Using adversarial training, we force the appearance of the resulting lane image to resemble that of the ground truth lane images.

Self-supervision leverages automatically generated supervision on auxiliary tasks related to the objective task to force the creation of useful intermediate network representations. To this end, we introduce a new self-supervision task, which improves performance on lane detection. We then use target domain self-supervision together with source domain supervised training for domain adaptation. Finally, we implemented three additional approaches that require some adjustment to the lane detection task: image-to-image translation [9], feature dis-

tribution alignment using adversarial learning [10] and central moment discrepancy [11]. We evaluate all methods on three datasets: tuSimple [3], llamas [12] and 3DLanes [13]. We show that applying domain adaptation, using several of the tested methods, significantly improves performance on the target domain, as can be observed in Figure 1(b). We also show that it improves performance in the semi-supervised setting. In particular, our autoencoder approach nearly closes the accuracy gap on two out of the three datasets compared to full supervision with only a tenth of the labels. To summarize, our main contributions are:

- Showing that synthetic-to-real domain adaptation can significantly improve lane detection performance when there is little or no target domain labels
- Introducing a novel autoencoder-based DA approach for lane detection that achieves state-of-the-art performance in the SSDA setting on all tested datasets
- Introducing a new self-supervision objective for lane detection
- Providing a comprehensive comparison and evaluation of proposed and existing methods in both UDA and SSDA settings.

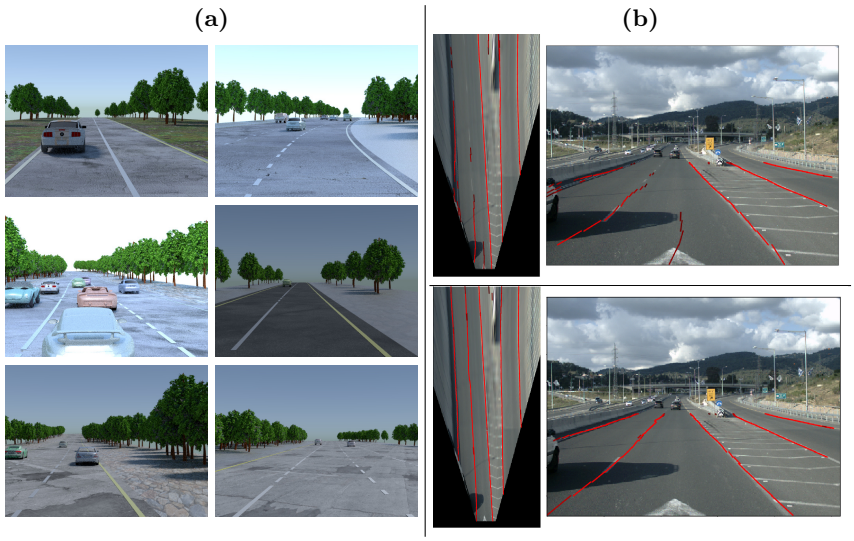

Fig. 1. (a) Examples of images from our source domain, consisting of synthetic images randomly generated using the methodology of [2]. **(b)** Lane detection results on an example from the 3DLanes dataset [13]. **Top:** result of a model trained with synthetic data without domain adaptation. **Bottom:** After domain adaptation using the self-supervision objective described in Section 3. Each result is shown in top-view, as obtained directly from our network, and back-projected to regular view on the right. The result is shown by highlighting in red the detected lane segments for tiles with confidence above a minimum threshold. The brightness of the segment color reflects its confidence score.

2 Related work

There has been extensive prior work on unsupervised domain adaptation. The general idea is to align the distributions of the source and target domains at some level of the representation. This can be accomplished by two general strategies. The first strategy attempts to align the two domains at an intermediate feature-level representation. One way to achieve this is to directly minimize the discrepancy between the two distributions using measures such as maximum mean discrepancy (MMD) [14, 15] and central moment distance (CMD) [11]. Another approach for bringing representations closer is based on adversarial learning [10, 16]. The idea is to train an adversarial discriminator that distinguishes between the two domains, encouraging the generator, in this case, the feature embedding network, to generate indistinguishable feature maps. Feature embedding weights may be shared between the domains [10] or kept separated [16]. Several additional variants of this approach have been proposed: [17] condition the adversarial game not only on the embedding but also on the final output, [18] align the features at multiple levels, and [19] use the discriminator over images generated from the embedding. Our implementation for this approach (**Embedding GAN**) uses a adversarial learning on the feature embedding and shared weights as in [10].

The second strategy uses image-to-image translation to impose alignment at the raw image level. Image translation methods [20, 21] learn the mapping between two image domains, and is mainly used for style transfer, object transfiguration, season transfer, and photo enhancement. In the context of domain adaptation, the image translation generates a target domain image from a raw source domain one, allowing to train a model in the supervised setting on a "target-like" dataset [22, 9, 23, 24]. As shown in [9], the two strategies can be combined to improve performance further. For our task we implement **image translation** using CycleGAN [20].

Self-supervised learning uses an auxiliary task generated automatically from the data itself to train feature representations that would hopefully be useful for the end-goal task. Many auxiliary tasks have been proposed, such as jigsaw puzzle solving [25], image rotation prediction [26], and contrastive predictive coding [27]. Typically, self-supervision is applied to large sets of unlabeled data as a pre-train stage before supervised training [28]. Naturally, as concurrently proposed by [29], self-supervision can be used for unsupervised domain adaptation. The idea is to train the feature embedding on the supervised task using the labeled source dataset, and on the auxiliary task using the unlabeled target images. Assuming the correlation between the tasks, the hope is that the feature embedding will encode the correlated information similarly for both domains. In [29], the feature embedding is further aligned by training the self-supervision task on the source domain.

Compared to the research in domain adaptation for *classification*, much less attention has been paid to domain adaptation for other, more complex, computer vision tasks. Several studies address object detection [30, 31], monocular depth estimation [32, 33] and semantic segmentation [9, 34–36, 24]. To our best

knowledge, domain adaptation for lane detection has yet to be addressed. While synthetic-to-real adaptation for automotive perception has been previously studied [37, 9, 34–36], it was always with manually created synthetic datasets (e.g. vKITTI [7], SYNTHIA [38]) as opposed to the random generation approach we adopt from [2].

3 Methods

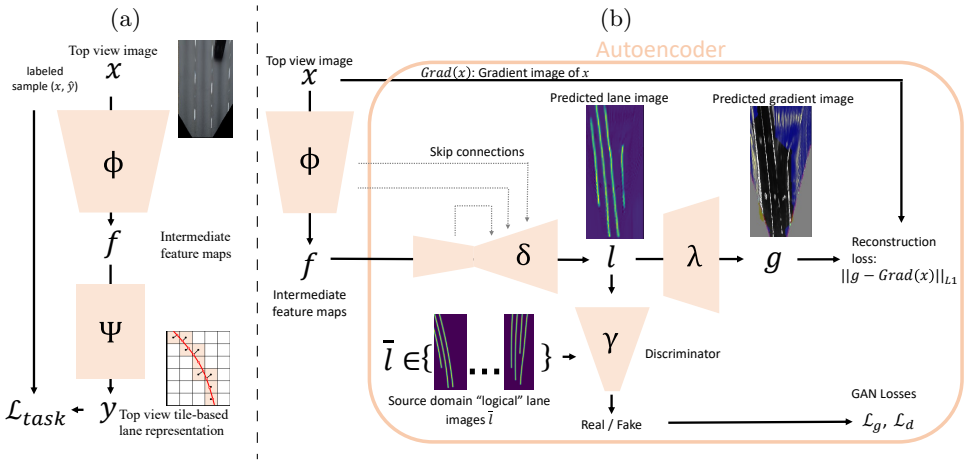

Fig. 2. (a) Base architecture for lane detection. $f = \phi(x)$ are intermediate feature maps computed from the top view. $y = \psi(f)$ represents the detected lanes in the tiles representation [13]. This base architecture, used also in inference, is trained end to end in the fully supervised setting. **(b) Our proposed autoencoder architecture for training with unlabeled examples and unpaired top view lane images.** The networks ϕ, δ, λ are trained to minimize the reconstruction loss while $\delta \circ \phi$ and γ are adverserially trained as a generator and discriminator respectively, driving the generated lane image l to resemble ground truth annotations $\bar{l}$. Note that $\delta \circ \phi$ forms an hourglass architecture including skip connections. In g the blue pixels are not considered in the reconstruction loss using the heuristics described in Section 3. At inference the network can output $\delta \circ \phi(x)$, which may be additionally transformed to the tile representation using a pretrained transformer network (See Section 4 for more details).

We start by warping the original image I to a virtual top view image $x \in \mathcal{X} = \mathbb{R}^{3 \times H' \times W'}$ using an Inverse Perspective Mapping (IPM), which is a homography defined by the camera's position relative to the local ground plane, its intrinsic parameters and additional parameters embodied by H' and W'. Our lane detection network gets x and outputs a lane representation y describing the lanes in top view. Working in top view has the advantage of translation invariance, an important property for convolutional networks, and is also

beneficial to subsequent modules such as lane clustering and tracking [2, 13]. We use the semi-local tile-based representation, recently proposed in [13], that divides the top view into $H \times W$ non-overlapping tiles, each roughly corresponding 1.6 by 1.6 meters in the real world. For each tile (i, j) the network outputs the confidence ($b_{(i,j)} \in [0, 1]$) that there is a lane passing through the tile, and regresses using a set of continuous outputs $\mathbf{p}_{(i,j)} \in \mathbb{R}^9$ its position and orientation relative to the rectangle's center, assuming that locally the lane is linear. We follow the exact representation of [13] except for omitting the elevation offset which is used for detection in 3D. The output is then expressed as: $y = \{(b_{(i,j)}, \mathbf{p}_{(i,j)}) | i \in [1, \ldots, H], j \in [1, \ldots, W]\} \in \mathcal{Y} = R^{10 \times H \times W}$.

Our base network architecture, used in inference, is illustrated in Figure 2(a). It is convenient to present the base network as composed of two stages. First, an embedding convolutional neural network $\phi : \mathcal{X} \mapsto \mathcal{F} = \mathbb{R}^{C \times H \times W}$ generates an intermediate feature map representation f, with the same spatial dimensions as the output, and C channels. Then, an additional convolutional network $\psi : \mathcal{F} \mapsto \mathcal{Y}$ computes the output y from the intermediate feature maps f. Given a dataset $\mathcal{D}^l$ consisting of labeled examples $(x, \hat{y}) \in \mathcal{X}, \mathcal{Y}$, the supervised task function loss is:

$$\mathcal{L}_{task} = \sum_{(x,\hat{y}) \in \mathcal{D}^l} \mathcal{L}_{tiles}(\psi \circ \phi(x), \hat{y})$$

Where $\mathcal{L}_{tiles}(y, \hat{y})$ sums the loss across all tiles for a single output y and ground truth $\hat{y}$ as described in [13] omitting the elevation component. Note also that $\hat{y}$ is obtained by projecting *image annotations* to top-view using the same IPM applied to the image.

In the **fully supervised** setting, we simply have a labeled dataset $\mathcal{D}^l$ and train the base network with the task loss function $\mathcal{L}_{task}$. In the **unsupervised domain adaptation** (UDA) setting, we have two datasets, a labeled source domain dataset $\mathcal{D}^l_S$, and an unlabeled set of images $\mathcal{D}^u_T$ from the target domain. Finally, in the **semi-supervised domain adaptation** (SSDA) setting we have in addition, compared to the UDA setting, a small set of labeled target domain images $\mathcal{D}^l_T$. We next describe our proposed autoencoder (Section 3.1) and self-supervision (Section 3.2) approaches followed by a short description of the existing methods we additionally tested in Sections 3.3 and 3.4.

3.1 Autoencoder approach

Our proposed autoencoder approach is inspired by the human landmark detection approach of [8], in which a landmark detector is trained with unlabeled target domain images, and unpaired ground truth annotations. In our application, this approach is based on the task-specific assumption that in the road area lane markings correlate with image gradients. Essentially, we train a detector to generate, from an unlabeled input image x, a gray-scale "lane image" that satisfies two constraints: (1) it looks like a valid ground truth image of lanes and (2) it holds the information required to reconstruct the original gradients

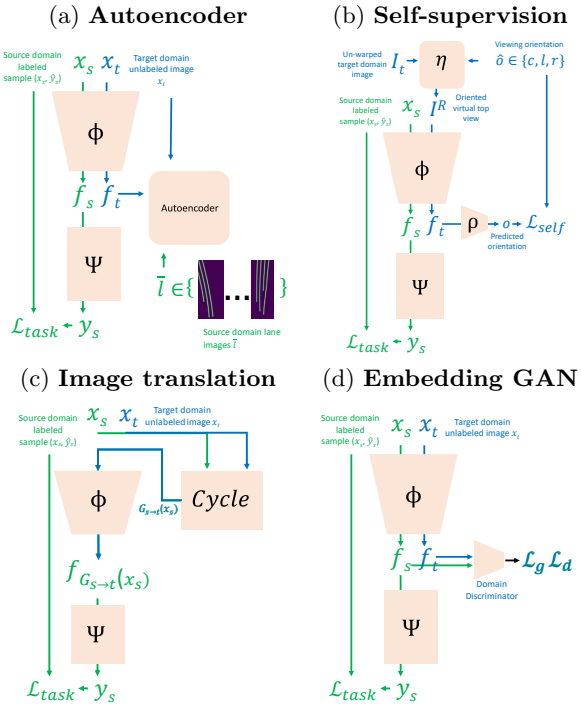

Fig. 3. Architectures for the different domain adaptation approaches in the UDA setting. See Section 3 for further details.

in the image. The first constraint is imposed by a discriminator, γ, which tries to distinguish between the generated lane image l and unpaired ground truth lane annotations. The second is imposed by a decoder that tries to reconstruct the original image gradients $Grad(x)$ from l. The entire approach, illustrated in Figure 2(b), relies on the assumption, that the natural candidate for the encoded image l satisfying both constraints is that of the lanes in the scene.

Formally, our training process gets as input the unlabeled target domain images $\mathcal{D}_T^u$ and a set of source domain ground truth "lane images", $\mathcal{D}_S^{up} = \{\bar{l}\}$, as described next. Given an input image $x \in D_T^u$ and its corresponding feature maps $f = \phi(x)$, an encoder, $\delta : \mathcal{F} \mapsto \mathcal{I}_L = \mathbb{R}^{W/4,H/4}$, generates a gray-scale "lane image" $l = \delta(f) \in \mathcal{I}_L$. The discriminator, γ, is trained to distinguish between the generated lane images and the ground truth lane images. The decoder λ, generates an image g, trained to reconstruct $Grad(x)$ using an L_1 loss:

$$\mathcal{L}_{reconstruct} = \sum_{x \in \mathcal{D}_T^u} \|\lambda \circ \delta \circ \phi(x) - Grad(x)\|_1$$

The GAN loss functions consist of the discriminator loss function $\mathcal{L}_d$, minimized for the discriminator (γ) parameters, and the generator loss function $\mathcal{L}_g$,

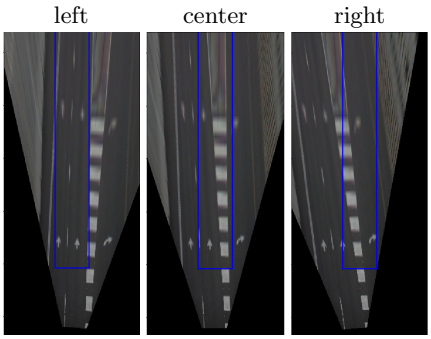

Fig. 4. Self-supervision task for lane detection. Depicted is the same image transformed for the right, center and left viewing directions. The network then gets the central rectangle (outlined in blue) and has to predict the viewing angle it was generated with.

minimized for the parameters of the generator which in our case is $\delta \circ \phi$. We chose $\mathcal{L}_g, \mathcal{L}_d$ following the Wasserstein GAN with gradient penalty (WGAN-GP) suggested by [39]. Training the network alternates between two objectives: minimizing the discriminator loss $\mathcal{L}_d$ over the parameters of the discriminator, γ, and minimizing a combination of the reconstruction and generator losses $\alpha L_{reconstruct} + L_g$ over the parameters of δ, ϕ and λ.

One hurdle to overcome in this approach is the simple fact that not all gradients in the top-view image $Grad(x)$ can be explained by the lane image, with edges belonging to other road markings, on-road objects, sidewalks and even the lane mark dashing themselves. We implement several heuristics to mitigate the effect of all other factors: we apply a car detector (YOLOv3 [40]) and mark the detected pixels to be ignored in the reconstruction loss. Similarly, we ignore everything outside a dilated convex hull around the lanes detected in l. Finally, as can be seen in Figure 2(b), the decoder, λ, manages to reconstruct other road markings and lane dash positions, by hiding appearance ques within l, a phenomena observed by [8] as well. This base architecture can be trained in an **unpaired labels** setting, i.e. without having any source domain images, but with only the set of unpaired ground truth lane images from the source domain, $\mathcal{D}_S^{up}$. While in this setting the method converged at times (See result example in the appendix), in general it is highly unstable. This instability was solved by adding direct task supervision using the source dataset in the UDA setting as described next.

In the UDA setting we additionally have source domain images paired with their labels, $\mathcal{D}_S^l$. The data from source domain has, thus, two roles while training - pairs of images and labels are used to minimize $\mathcal{L}_{task}$, and labels are also used to generate lane images $\bar{l}$ for the autoencoder adversarial learning. Figure 3(a) illustrates this UDA architecture. Notice that the feature embedding, ϕ, is shared between the two domains, and hence is the network in which domain adaptation

occurs. On the other hand, the high level reasoning network (ψ) producing the final lane result, sees only source domain images, and hence expects the feature maps f to be domain invariant. We follow this decomposition for all our tested domain adaptation methods.

3.2 Self-supervision by viewing orientation prediction

In self-supervision it is important to find an auxiliary task that will train a representation informative for the end-goal task. To this end, we introduce a new self-supervision task illustrated in Figure 4. The basic idea is to generate one of three viewing angles of the scene and let the network predict which one it is. Since the vehicle orientation is commonly aligned with the lanes, predicting the camera viewing direction is strongly related to the capability to correctly detect the lanes. Formally, each un-warped image I is transformed to a virtual top view according to a selected orientation $\hat{o}$: either the regular central one (c), or one of two additional views in which the camera is virtually panned by 5 degrees to the left (l) or to the right (r). We then crop from the resulting top-view a rectangular image from a fixed area in the center of the view. This orientation dependent transformation, η, is equivalent, up to image sampling differences, to cropping an oriented rectangle from the top-view image x. The self-supervision task is then to predict the orientation o from the cropped rectangle I_r. To this end we train a small classifier network ρ on top of the embedding features: $o = \rho \circ \phi \circ \eta(I, \hat{o})$. The self-supervision objective function $\mathcal{L}_{self}$ is the cross-entropy loss for the three-way classification task. In the UDA setting we train using the self-supervision objective by minimizing $\mathcal{L}_{self}$ for the unlabeled target images and $\mathcal{L}_{task}$ for the labeled source images, as illustrated in Figure 3.

3.3 Image translation

As proposed in the Cycada framework for UDA [9], we use CycleGAN [20] for image-to-image translation. Figure 3(c) illustrates the approach. Source images x_s are mapped to target images x_t. We slightly abuse notations for simplicity because here x_s, x_t are the original unwarped images, not in top-view. A discriminator γ_t and generator $G_{s\to t}$ are trained using the loss:

$$\mathcal{L}_{\text{GAN}}(G_{s\to t}, \gamma_t) = \min_{G_{s\to t}} \max_{\gamma_t} \left(\sum_{x_t \in \mathcal{D}_t^u} [\log \gamma_t (x_t)] + \sum_{x_s \in \mathcal{D}_s^l} [\log(1 - \gamma_t (G_{s\to t} (x_s))] \right)$$

A discriminator γ_s and generator $G_{t\to s}$ are trained analogously.

As in CycleGAN [20], cycle-consistency loss is used to ensure image content is preserved while translating the original image. This allows using the source labels y_s with the translated source images, $G_{s\to t}(x_s)$, to minimize the task loss, $\mathcal{L}_{task}$. We optimize the task loss in a second stage over the translated images (transformed to top-view). Combined single-stage training of the CycleGAN and task loss did not improve performance. We also tried to combine feature-level distribution matching as proposed in the Cycada framework [9], but did not gain any accuracy improvement.

3.4 Feature-level distribution matching

We chose implementing a GAN approach [10] and the Central Moment Discrepancy CMD [11] for feature-level distribution matching. The goal is to push the distribution of the feature map representations for the source domain (f_s) and the target domain (f_t) towards one another. In the **embedding GAN** framework, we train a discriminator to distinguish between the domain of different examples from the feature embedding and a generator, in our case the feature embedding function ϕ, to fool the discriminator. Again, we use the GAN loss functions $\mathcal{L}_g, \mathcal{L}_d$ following the WGAN-GP formulation [39]. In the UDA setting, $\mathcal{L}_{task}$ and $\mathcal{L}_g$ are simultaneously minimized over the source images. Figure 3(d) illustrates this approach. In the **CMD** approach the GAN is replaced by a distance loss on the first two central moments between the two distributions as in [11]. Details for the latter approach are brought in the appendix.

4 Experiments

We test the various approaches for domain adaptation on three different lane detection benchmarks. In each experiment, we set the synthetic dataset as the source domain, and one of the three benchmarks as the target domain. Each training session is run for a fixed number of iterations. In the semi-supervised setting, we choose a **labeled subset** of the training data (roughly 10%), consisting of one or several, separate driving sessions.

4.1 Datasets

Synthetic dataset. In all our experiments, we follow the methodology proposed in [2] to generate a synthetic dataset as the source domain. Images are generated by randomly drawing the parameters for each scene using the generation method from [2] (See examples in Figure 1(a)). What is unique about this methodology is its simplicity. As opposed to manually or semi-manually generated synthetic datasets such as [7, 38], this methodology uses a very small set of graphical assets and instead achieves scene variability by randomly varying lane topology and geometry. In this sense, it can be viewed as "free data" which can be generated using a simple algorithm and an open-source graphic engine. For each target domain, we generated 50K examples using the same scene parameters, but with camera intrinsic and extrinsic parameters roughly adjusted to the target domain. In our experiments generating more images did not result in accuracy gains. This may be due to the limited variability of the appearance in our synthetic data, or due to the limited variability of the lane geometry in the target domains.

tuSimple. The tuSimple lane dataset [3] consists of 3,626 training and 2,782 test images in mostly highway scenarios. While relatively small and not very diverse, we chose it for being the most studied. We report results on the test set.

llamas. The llamas lane-marker dataset [12] is a newer, much larger dataset, consisting of over 100K labeled images. Created from 14 highway recordings

of around 25 km each, together with high accuracy maps in a fully automatic process, it is one of the largest datasets available today. As labels are not available for the test set, we show the evaluation results on a validation set, which consists of a driving session not used during training.

3DLanes. The 3DLanes [13] dataset is the most recent and most diverse dataset among the three. It contains 330000 images labeled in a semi-automated process, in highway and rural environments. Again, we report results on the validation set.

4.2 Evaluation Metrics

Segment-based evaluation. In [13], evaluation is preformed only after lane-level clustering. In this study, the goal is to compare the different DA methods directly, and therefore we refrain from applying post-processing methods that can skew the results. For this purpose, we propose a direct tile-representation evaluation formulation. The segment-based evaluation follows the principles of object detection evaluation, where detected lane segments are matched to ground truth ones, and Precision-Recall curves are computed for different matching criteria.

For a single image, the input for the evaluation is an unordered set SEG_{out} comprising of all output lane segments $\mathbf{p}$ (with corresponding detection confidence score b), and the set of all ground truth lane segments obtained using the same representation $SEG_{gt} = \{\hat{\mathbf{p}}\}$. We first compute a symmetric lane segment distance $seg\_dist(\mathbf{p}, \hat{\mathbf{p}})$ for all segment pairs ($\mathbf{p} \in SEG_{out}, \hat{\mathbf{p}} \in SEG_{gt}$), reflecting a geometric distance between the segments in the top view (see appendix for a detailed formulation of seg_dist). We then use the Hungarian algorithm [41] to find a minimum distance matching between the two sets of lane segments. Given a maximum distance seg_dist_{max} (analogous to maximum IoU in object detection), We compute the Recall-Precision (RP) curve, considering all segment matches (in all the test images) with $seg\_dist < seg\_dist_{max}$, by iterating over the confidence values (b) of the detected segments. Each RP curve is summarized as the Average Precision (AP). Our final evaluation metric, the mean Average Precision (mAP), is computed as the the average AP for five different maximum matching distances: $seg\_dist_{max} \in \{10cm, 20cm, 30cm, 40cm, 50cm\}$. These distances correspond to real-world distances in the road plane.

Lane-based evaluation. For completeness, we also compute the lane-based evaluation metric used in the tuSimple benchmark [3]. This metric requires a single detection per lane, and hence further post-processing, namely clustering of output tiles. To this end, we deploy a simple, heuristic clustering algorithm described in the appendix. In this study, there are several disadvantages to using it. A partial list includes:

- it is indirect by relying on an additional component - the clustering
- it evaluates performance in the image plane and not in top view as our method outputs
- it assumes a known number of lanes in the scene

– we observed that in practice that it is less correlated with performance of the detector

4.3 Implementation details

All the implemented modules are convolution neural networks. The feature embedding function, ϕ, is based on the VGG architecture [42]. All experiments were run from random initialization for a pre-defined number of iterations. We use the ADAM optimizer [43] without weight decay. For most methods we observed a non-negligible variability in performance between training iterations, even towards the end of each training session. To reduce the effect of this phenomena, all the results we report are averaged over 5 different snapshots 100 iterations apart at the end of the session. The remaining architecture and training protocol details are provided in the appendix.

4.4 Results

Table 1 summarizes our results on each of the three datasets. The first row shows the **fully supervised** result, which serves as the upper bound for all other tested methods. Compared with state-of-the-art (96.6%, [1]) our base supervised method without any bells and whistles reaches 95.1% on the tuSimple benchmark using the tuSimple lane-based evaluation metrics. All the results in Table 1 use our mAP metric. Apparent from the experiments is that the llamas and 3DLanes datasets are indeed more difficult compared to the tuSimple. The **synthetic only** model is trained with the synthetic labeled data without performing any adaptation to the target domain. Of all tested methods, it gives the poorest results due to the noticeable, significant domain gap that can be observed in Figure 1. This model serves as the baseline for all UDA methods.

Unsupervised domain adaptation. In the UDA setting, The baseline-gap (BG) column, specifies the portion of the accuracy gap between the **fully supervised** and the **synthetic only** models, closed by the corresponding method. From the five domain adaptation method tested, three methods, namely **self-supervision (S)**, **image translation (IT)** and **autoencoder (AE)**, gained significant improvements over the non-adapted baseline (**synthetic only**). The remaining two methods, **CMD** and **embedding GAN** gave worse results in most experiments. Among the three leading methods, performance is similar, with slight advantages to one over the other in different experiments. We also tried all combinations of these three methods, with some combinations bringing small additional improvements. Details on the training strategy for combined domain adaptation methods are in the appendix. On the tuSimple and 3DLanes benchmarks the **image-translation** method was the single best performing method while on llamas **self-supervision** performed best. different combinations of the three leading methods seem to further improve performance, and combining all three (**AE+IT+S**), performed best on two out of the three datasets closing more than 70% of the baseline-gap. Notably, in all experiments,

training with self-supervision was more stable and reproducible compared with the two competing methods that rely on adversarial training.

Semi-supervised. In this setting we have roughly 10% of the target domain labels. We start with the simplest option, **small supervision**, in which we train a supervised model using this small portion of target domain data. Interestingly, adding labeled synthetic data (**small+syn. supervision**) to the train set, significantly improves performance for all three datasets even without domain adaptation. The latter serves as the baseline for all semi-supervised domain adaptation methods, and used to compute the BG measure. In contrast with the other methods, self-supervision does not require source domain data, but can exploit unlabeled target domain images. Our experiments show that this method, **self-sup. w/o syn**, improves accuracy in two out of the three benchmarks.

Semi-supervised domain adaptation As in the UDA setting, we evaluated all domain adaptation methods and combinations, but this time with an extra portion of labeled target domain data. As in the UDA setting we observed inferior performance for the **CMD** and the **embedding GAN** methods, omitted in Table 1 for brevity. As expected the additional labeled data improves the resulting accuracy for each method compared to the UDA setting. Our proposed autoencoder approach outperforms all other single methods on all datasets and all method combinations except for one - **AE+S** on llamas. On tuSimple and llamas the autoencoder approach respectively delivers 1.8% and 2.7% mAP less than the fully supervised model. The practical implication of this result is that using our approach, sacrificing this small loss in accuracy provides a ten fold saving in data annotation. We also note that each experiment we present requires a significant computational effort, and therefore a systematic study of semi-supervision with different amounts of labeling is beyond the scope of this study.

5 Conclusions

We showed that it is possible to improve lane detection when labeled data availability is limited using domain adaptation from random synthetic data. To this end we introduced a new autoencoder approach for domain adaptation and a new task-specific self-supervision objective. We further explored the different existing domain adaptation approaches and their effectiveness for the lane detection task. While these findings have practical implications in autonomous driving research, we suggest further study beyond the scope of this work: experimenting with varying amounts of supervised data in the semi-supervised setting, exploring the effect of the complexity of the synthetic data on the final results, and adapting the detector for lanes in *3D*.

		%lbl. data	Syn. data	tuSimple mAP%	tuSimple BG%	llamas mAP%	llamas BG%	3DLanes mAP%	3DLanes BG%
	fully supervised	100	-	81.1	100	73.2	100	74.5	100
	synthetic only	-	✓	60.5	0	47.8	0	53.8	0
UDA	Autoencoder	-	✓	67.7	35.0	56.0	32.3	57.8	19.1
UDA	Image translation	-	✓	**72.0**	**55.7**	62.3	57.3	**59.3**	**26.5**
UDA	Self supervision	-	✓	66.3	27.9	**63.4**	**61.6**	58.1	20.5
UDA	CMD	-	✓	62.7	10.6	51.9	15.9	55.7	9.0
UDA	Embedding GAN	-	✓	52.1	0	37.8	0	31.9	0
UDA	AE+IT	-	✓	73.4	62.5	65.6	70.0	59.6	27.8
UDA	AE+S	-	✓	70.1	46.3	63.7	62.6	59.7	28.7
UDA	IT+S	-	✓	72.4	57.8	65.3	69.1	**60.1**	**30.6**
UDA	AE+IT+S	-	✓	**77.1**	**80.5**	**66.0**	**71.6**	59.5	27.5
	small supervision	10	-	74.4	-	60.1	-	56.9	-
	self-sup. w/o syn	10	-	75.2	-	68.6	-	60.1	-
	small+syn. supervision	10	✓	77.5	0	65.5	0	62.0	0
SSDA	Autoencoder	10	✓	**79.3**	**50.0**	**70.5**	**65.6**	**66.9**	**39.0**
SSDA	Image translation	10	✓	78.4	25.0	69.1	47.1	64.2	17.5
SSDA	Self supervision	10	✓	77.1	0	70.2	61.1	64.0	16.0
SSDA	AE+IT	10	✓	78.8	37.7	70.3	63.4	66.2	33.3
SSDA	AE+S	10	✓	77.6	4.6	**70.7**	**68.3**	65.2	25.6
SSDA	IT+S	10	✓	77.8	8.9	70.1	59.6	65.8	30.4
SSDA	AE+IT+S	10	✓	77.7	5.2	70.4	64.5	66.6	36.7

Table 1. Results on the tuSimple [3], llamas [12] and 3DLanes [2] datasets using the mean Average Precision (mAP) using the **segment-based** evaluation. Values in the column "**%lbl. data**" correspond to the percent of labeled data *from the respective target domain* used by each method. The "**Syn. data**" column specifies whether synthetic data was used in training. **Unsupervised domain adaptation:** rows in the section marked **UDA** correspond to unsupervised domain adaptation experiments. The BG% (Baseline Gap) for these methods measures the percent of the gap closed between the **synthetic only** and the **fully supervised** mAP: BG(**method**)=(mAP(**method**)-mAP(**synthetic only**))/(mAP(**fully supervised**)-mAP(**synthetic only**)). The best single method and best method combination are marked in bold. **Semi-supervised domain adaptation:** rows in the section marked **SSDA** correspond to semi-supervised domain adaptation experiments. The BG% (Baseline Gap) for these methods measures the percent of the gap closed between the **small+syn. supervision** and the **fully supervised** mAP: BG(**method**)=(mAP(**method**)-mAP(**small+syn. supervision**))/(mAP(**fully supervised**)-mAP(**small+syn. supervision**)). The best single method and best method combination are marked in bold.

References

1. Hou, Y., Ma, Z., Liu, C., Loy, C.C.: Learning lightweight lane detection cnns by self attention distillation. In: ICCV. (2019) 1013–1021
2. Garnett, N., Cohen, R., Pe'er, T., Lahav, R., Levi, D.: 3d-lanenet: end-to-end 3d multiple lane detection. In: ICCV. (2019) 2921–2930
3. : (`http://benchmark.tusimple.ai`, lane challenge)
4. Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., Vasudevan, V., Han, W., Ngiam, J., Zhao, H., Timofeev, A., Ettinger, S., Krivokon, M., Gao, A., Joshi, A., Zhang, Y., Shlens, J., Chen, Z., Anguelov, D.: Scalability in perception for autonomous driving: Waymo open dataset. CoRR **abs/1912.04838** (2019)
5. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. arXiv preprint arXiv:1903.11027 (2019)
6. Gaidon, A., Wang, Q., Cabon, Y., Vig, E.: Virtual worlds as proxy for multi-object tracking analysis. In: CVPR. (2016) 4340–4349
7. Gaidon, A., Wang, Q., Cabon, Y., Vig, E.: Virtualworlds as proxy for multi-object tracking analysis. In: CVPR, IEEE Computer Society (2016) 4340–4349
8. Jakab, T., Gupta, A., Bilen, H., Vedaldi, A.: Learning landmarks from unaligned data using image translation. CoRR **abs/1907.02055** (2019)
9. Hoffman, J., Tzeng, E., Park, T., Zhu, J.Y., Isola, P., Saenko, K., Efros, A.A., Darrell, T.: Cycada: Cycle consistent adversarial domain adaptation. In: ICML. (2018)
10. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: ICML. Volume 37 of ICML'15., JMLR.org (2015) 1180–1189
11. Zellinger, W., Grubinger, T., Lughofer, E., Natschläger, T., Saminger-Platz, S.: Central moment discrepancy (CMD) for domain-invariant representation learning. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings, OpenReview.net (2017)
12. Behrendt, K., Soussan, R.: Unsupervised labeled lane markers using maps. In: ICCV. (2019)
13. Efrat, N., Bluvstein, M., Garnett, N., Levi, D., Oron, S., Shlomo, B.E.: Semi-local 3d lane detection and uncertainty estimation. CoRR **abs/2003.05257** (2020)
14. Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., Erhan, D.: Domain separation networks. In: NIPS. Curran Associates, Inc. (2016) 343–351
15. Long, M., Cao, Y., Wang, J., Jordan, M.I.: Learning transferable features with deep adaptation networks. In: ICML. Volume 37 of ICML'15., JMLR.org (2015) 97–105
16. Tzeng, E., Hoffman, J., Darrell, T., Saenko, K.: Adversarial discriminative domain adaptation. In: CVPR. (2017)
17. Long, M., CAO, Z., Wang, J., Jordan, M.I.: Conditional adversarial domain adaptation. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R., eds.: NIPS. (2018) 1640–1650
18. Lahiri, A., Agarwalla, A., Biswas, P.K.: Unsupervised domain adaptation for learning eye gaze from a million synthetic images: An adversarial approach. CoRR **abs/1810.07926** (2018)
19. Sankaranarayanan, S., Balaji, Y., Castillo, C.D., Chellappa, R.: Generate to adapt: Aligning domains using generative adversarial networks. CoRR **abs/1704.01705** (2017)

20. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: ICCV. (2017) 2223–2232
21. Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: ICCV. (2017) 1501–1510
22. Taigman, Y., Polyak, A., Wolf, L.: Unsupervised cross-domain image generation. In: ICLR, OpenReview.net (2017)
23. Bousmalis, K., Silberman, N., Dohan, D., Erhan, D., Krishnan, D.: Unsupervised pixel-level domain adaptation with generative adversarial networks. In: CVPR, IEEE Computer Society (2017) 95–104
24. Chen, Y.C., Lin, Y.Y., Yang, M.H., Huang, J.B.: Crdoco: Pixel-level domain transfer with cross-domain consistency. (2019)
25. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: ECCV, Springer (2016) 69–84
26. Gidaris, S., Singh, P., Komodakis, N.: Unsupervised representation learning by predicting image rotations. In: ICLR. (2018)
27. van den Oord, A., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. CoRR **abs/1807.03748** (2018)
28. Hénaff, O.J., Razavi, A., Doersch, C., Eslami, S.M.A., van den Oord, A.: Data-efficient image recognition with contrastive predictive coding. CoRR **abs/1905.09272** (2019)
29. Sun, Y., Tzeng, E., Darrell, T., Efros, A.A.: Unsupervised domain adaptation through self-supervision. CoRR **abs/1909.11825** (2019)
30. Chen, Y., Li, W., Sakaridis, C., Dai, D., Van Gool, L.: Domain adaptive faster r-cnn for object detection in the wild. In: CVPR. (2018)
31. Zhu, X., Pang, J., Yang, C., Shi, J., Lin, D.: Adapting object detectors via selective cross-domain alignment. In: CVPR. (2019) 687–696
32. Abarghouei, A.A., Breckon, T.P.: Real-time monocular depth estimation using synthetic data with domain adaptation via image style transfer. In: CVPR 2018, IEEE Computer Society (2018) 2800–2810
33. Zheng, C., Cham, T.J., Cai, J.: T2net: Synthetic-to-realistic translation for solving single-image depth estimation tasks. In: ECCV. (2018) 767–783
34. Hong, W., Wang, Z., Yang, M., Yuan, J.: Conditional generative adversarial network for structured domain adaptation. In: CVPR. (2018) 1335–1344
35. Zhang, Y., Qiu, Z., Yao, T., Liu, D., Mei, T.: Fully convolutional adaptation networks for semantic segmentation. Proceedings of CVPR (2018)
36. Sankaranarayanan, S., Balaji, Y., Jain, A., Lim, S., Chellappa, R.: Learning from synthetic data: Addressing domain shift for semantic segmentation. In: CVPR, IEEE Computer Society (2018) 3752–3761
37. Chen, Y., Li, W., Chen, X., Van Gool, L.: Learning semantic segmentation from synthetic data: A geometrically guided input-output adaptation approach. In: CVPR. (2019) 1841–1850
38. Ros, G., Sellart, L., Materzynska, J., Vazquez, D., Lopez, A.M.: The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In: CVPR. (2016) 3234–3243
39. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. In: NIPS. (2017) 5767–5777
40. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. CoRR **abs/1804.02767** (2018)
41. Kuhn, H.W.: The hungarian method for the assignment problem. Naval research logistics quarterly **2** (1955) 83–97

42. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR **abs/1409.1556** (2014)
43. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
44. Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y.: Spectral normalization for generative adversarial networks. In: International Conference on Learning Representations. (2018)
45. Salimans, T., Goodfellow, I.J., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. CoRR **abs/1606.03498** (2016)